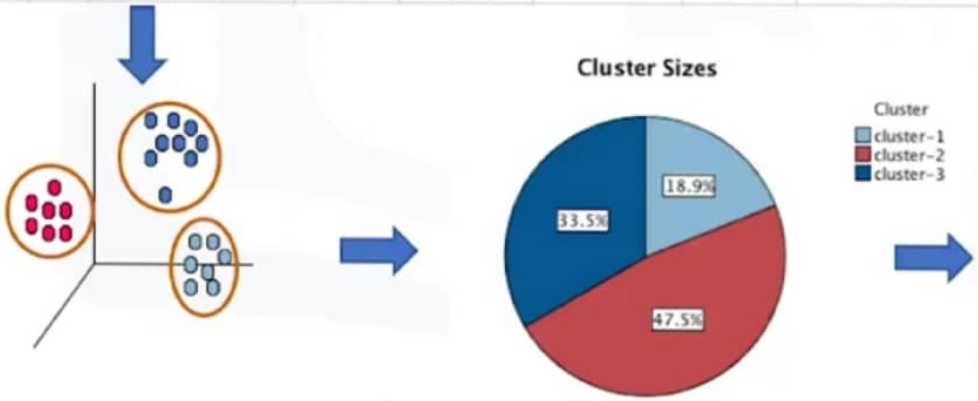
Clustering for segmentation

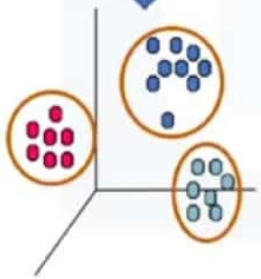
Customer Id	Age	Edu	Years Employed	Income	Card Debt	Other Debt	Address	DebtIncomeRatio	Defaulted
1	41	2	6	19	0.124	1.073	NBA001	6.3	0
2	47	1	26	100	4.582	8.218	NBA021	12.8	0
3	33	2	10	57	6.111	5.802	NBA013	20.9	1
4	29	2	4	19	0.681	0.516	NBA009	6.3	0
5	47	1	31	253	9.308	8.908	NBA008	7.2	0
6	40	1	23	81	0.998	7.831	NBA016	10.9	1
7	38	2	4	56	0.442	0.454	NBA013	1.6	0
8	42	3	0	64	0.279	3.945	NBA009	6.6	0
9	26	1	5	18	0.575	2.215	NBA006	15.5	1

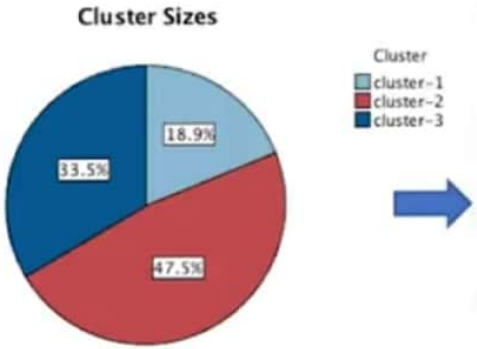


Cluster	Segment Name
cluster-1	AFFULUENT AND MIDDLE AGED
cluster-2	YOUNG EDUCATED AND MIDDLE INCOME
cluster-3	YOUNG AND LOW INCOME

Clustering for segmentation

Customer Id	Age	Edu	Years Employed	Income	Card Debt	Other Debt	Address	DebtIncomeRatio	Defaulted
1	41	2	6	19	0.124	1.073	NBA001	6.3	0
2	47	1	26	100	4.582	8.218	NBA021	12.8	0
3	33	2	10	57	6.111	5.802	NBA013	20.9	1
4	29	2	4	19	0.681	0.516	NBA009	6.3	0
5	47	1	31	253	9.308	8.908	NBA008	7.2	0
6	40	1	23	81	0.998	7.831	NBA016	10.9	1
7	38	2	4	56	0.442	0.454	NBA013	1.6	0
8	42	3	0	64	0.279	3.945	NBA009	6.6	0
9	26	1	5	18	0.575	2.215	NBA006	15.5	1





Customer ID	Segment
1	YOUNG AND LOW INCOME
2	AFFULUENT AND MIDDLE AGED
3	AFFULUENT AND MIDDLE AGED
4	YOUNG AND LOW INCOME
5	AFFULUENT AND MIDDLE AGED
6	AFFULUENT AND MIDDLE AGED
7	YOUNG AND LOW INCOME
8	YOUNG AND LOW INCOME
9	AFFULUENT AND MIDDLE AGED

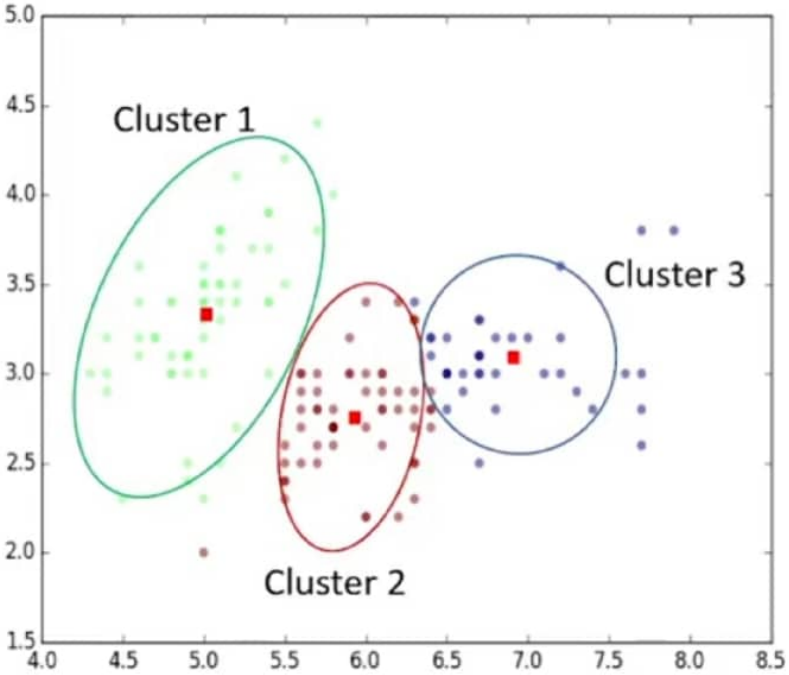
Cluster	Segment Name
cluster-1	AFFULUENT AND MIDDLE AGED
cluster-2	YOUNG EDUCATED AND MIDDLE INCOME
cluster-3	YOUNG AND LOW INCOME

individual customers preferences and their buying behaviors across various products.

What is clustering?

What is a cluster?

A group of objects that are **similar to other objects** in the cluster, and **dissimilar to data points** in other clusters.



Clustering Vs. classification

Labeled dataset

Customer Id	Age	Edu	Years Employed	Income	Card Debt	Other Debt	Address	DebtIncomeRatio	Defaulted
1	41	2	6	19	0.124	1.073	NBA001	6.3	0
2	47	1	26	100	4.582	8.218	NBA021	12.8	0
3	33	2	10	57	6.111	5.802	NBA013	20.9	1
4	29	2	4	19	0.681	0.516	NBA009	6.3	0
5	47	1	31	253	9.308	8.908	NBA008	7.2	0
6	40	1	23	81	0.998	7.831	NBA016	10.9	1
7	38	2	4	56	0.442	0.454	NBA013	1.6	0
8	42	3	0	64	0.279	3.945	NBA009	6.6	0
9	26	1	5	18	0.575	2.215	NBA006	15.5	1
10	47	3	23	115	0.653	3.947	NBA011	4	0

Modeling

Decision Tree

Prediction

Unlabeled dataset

Customer Id	Age	Edu	Years Employed	Income	Card Debt	Other Debt	Address	DebtIncomeRatio	Defaulted
1	41	2	6	19	0.124	1.073	NBA001	6.3	0
2	47	1	26	100	4.582	8.218	NBA021	12.8	0
3	33	2	10	57	6.111	5.802	NBA013	20.9	1
4	29	2	4	19	0.681	0.516	NBA009	6.3	0
5	47	1	31	253	9.308	8.908	NBA008	7.2	0
6	40	1	23	81	0.998	7.831	NBA016	10.9	1
7	38	2	4	56	0.442	0.454	NBA013	1.6	0
8	42	3	0	64	0.279	3.945	NBA009	6.6	0
9	26	1	5	18	0.575	2.215	NBA006	15.5	1

Modeling

Segmentation

K-Means

Clustering applications

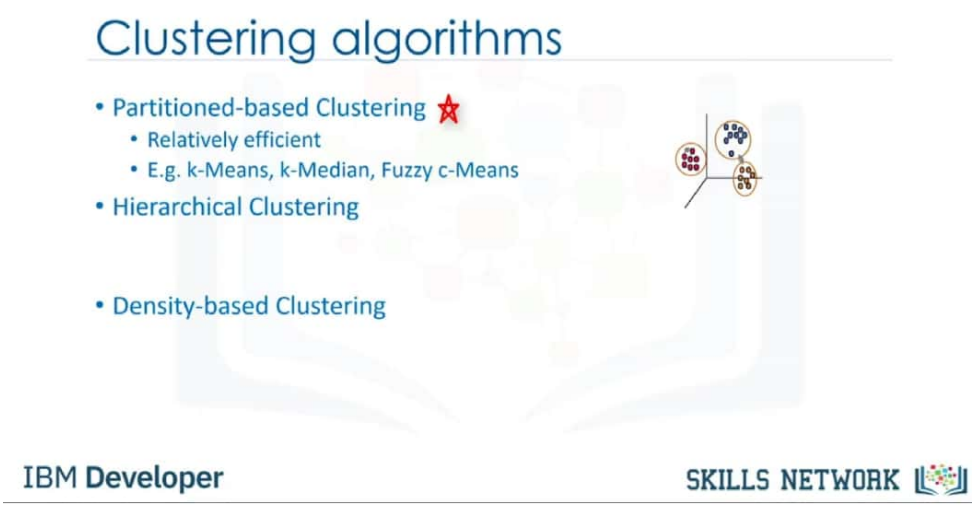
- RETAIL/MARKETING:
  - Identifying buying patterns of customers
  - Recommending new books or movies to new customers
- BANKING:
  - Fraud detection in credit card use
  - Identifying clusters of customers (e.g., loyal)
- INSURANCE:
  - Fraud detection in claims analysis
  - Insurance risk of customers

Clustering applications

- PUBLICATION:
  - Auto-categorizing news based on their content
  - Recommending similar news articles
- MEDICINE:
  - Characterizing patient behavior
- BIOLOGY:
  - Clustering genetic markers to identify family ties

Why clustering?

- Exploratory data analysis
- Summary generation
- Outlier detection
- Finding duplicates
- Pre-processing step

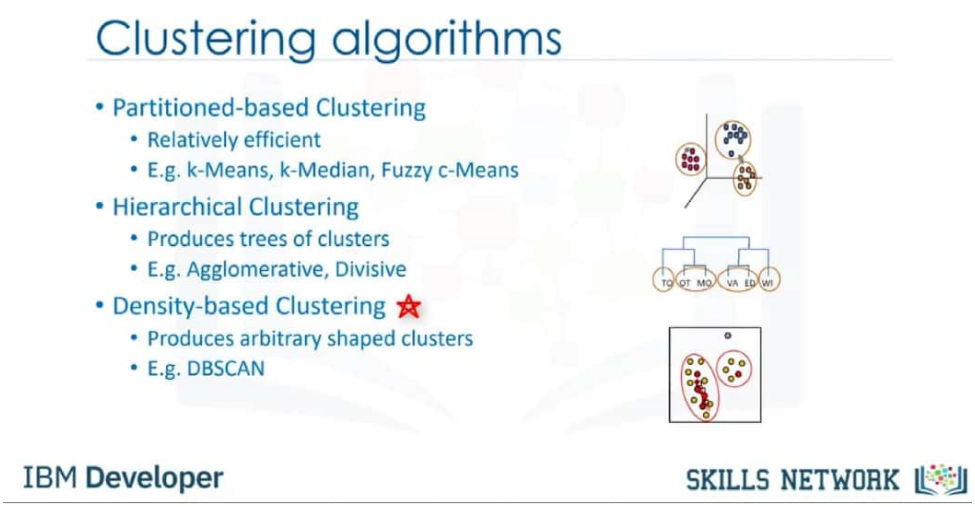


or as part of a complex system.

Let's briefly look at different clustering algorithms and their characteristics. Partition-based clustering is a group of clustering algorithms that produces sphere-like clusters, such as; K-Means, K-Medians or Fuzzy c-Means.

These algorithms are relatively efficient and are used for medium and large sized databases. Hierarchical clustering algorithms produce trees of clusters, such as agglomerative and divisive algorithms.

This group of algorithms are very intuitive and are generally good for use with small size datasets. Density-based clustering algorithms produce arbitrary shaped clusters. They are especially good when dealing with spatial clusters or when there



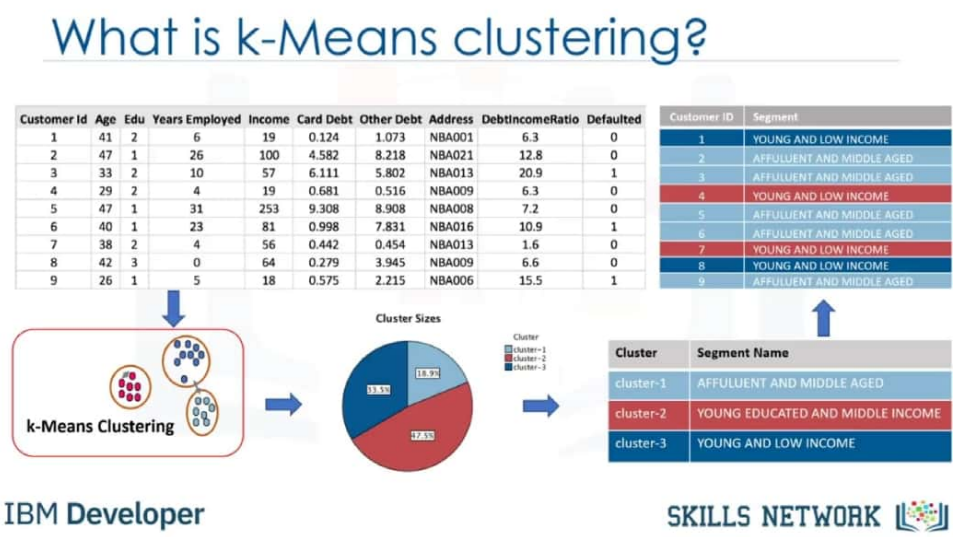
of clustering algorithms that produces sphere-like clusters, such as; K-Means, K-Medians or Fuzzy c-Means.

These algorithms are relatively efficient and are used for medium and large sized databases. Hierarchical clustering algorithms produce trees of clusters, such as agglomerative and divisive algorithms.

This group of algorithms are very intuitive and are generally good for use with small size datasets. Density-based clustering algorithms produce arbitrary shaped clusters. They are especially good when dealing with spatial clusters or when there is noise in your data set.

For example, the DB scan algorithm. This concludes our video. Thanks for watching! (Music)

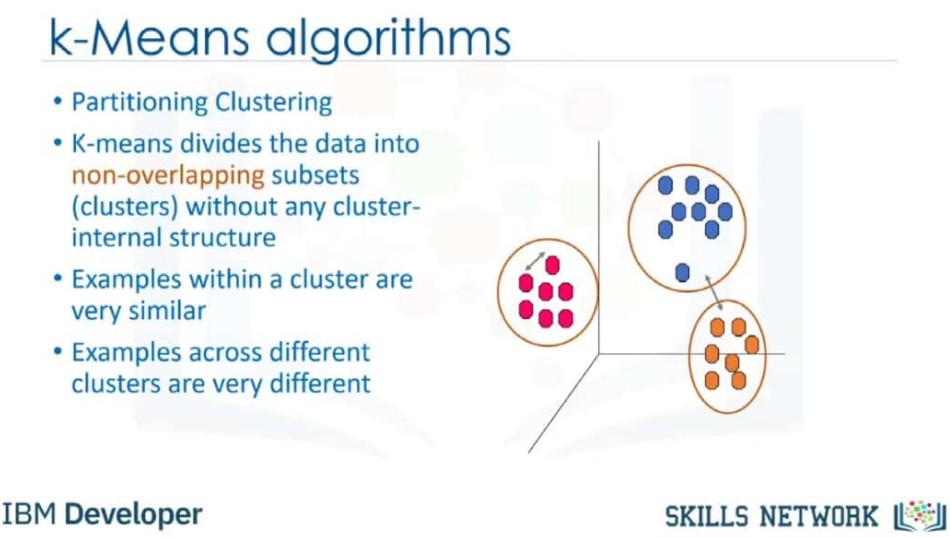
Intro to k-Means



Hello and welcome. In this video, we'll be covering K-Means Clustering. So let's get started. Imagine that you have a customer dataset and you need to apply customer segmentation on this historical data.

Customer segmentation is the practice of partitioning a customer base into groups of individuals that have similar characteristics. One of the algorithms that can be used for customer segmentation is K-Means clustering.

K-Means can group data only unsupervised based on the similarity of customers to each other. Let's define this technique more formally. There are various types of clustering algorithms such as partitioning, hierarchical or density-based clustering.

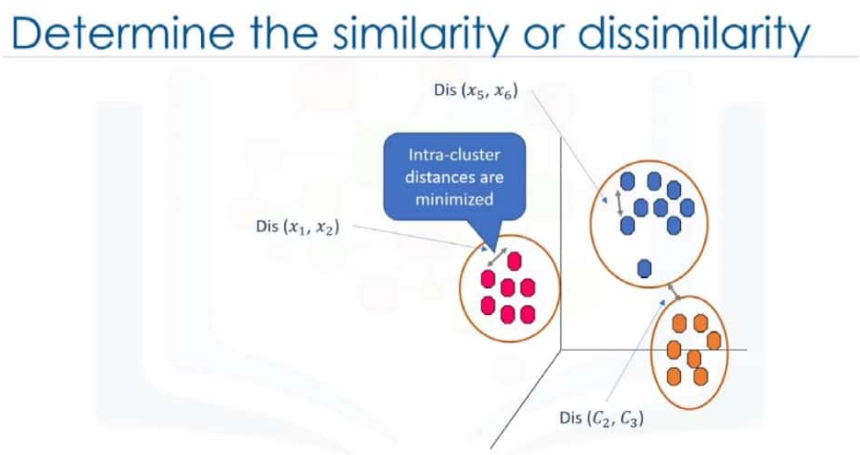


unsupervised based on the similarity of customers to each other. Let's define this technique more formally. There are various types of clustering algorithms such as partitioning, hierarchical or density-based clustering.

K-Means is a type of partitioning clustering, that is, it divides the data into K non-overlapping subsets or clusters without any cluster internal structure or labels. This means, it's an unsupervised algorithm.

Objects within a cluster are very similar, and objects across different clusters are very different or dissimilar. As you can see, for using K-Means we have to find similar samples: for example, similar customers. Now, we face a couple of key questions.

Intro to k-Means



First, how can we find the similarity of samples in clustering, and then how do we measure how similar two customers are with regard to their demographics?

Though the objective of K-Means is to form clusters in such a way that similar samples go into a cluster, and dissimilar samples fall into different clusters, it can be shown that instead of a similarity metric, we can use dissimilarity metrics.

In other words, conventionally the distance of samples from each other is used to shape the clusters. So we can say K-Means tries to minimize the intra-cluster distances and maximize the inter-cluster distances.

Now, the question is, how can we calculate the dissimilarity or distance

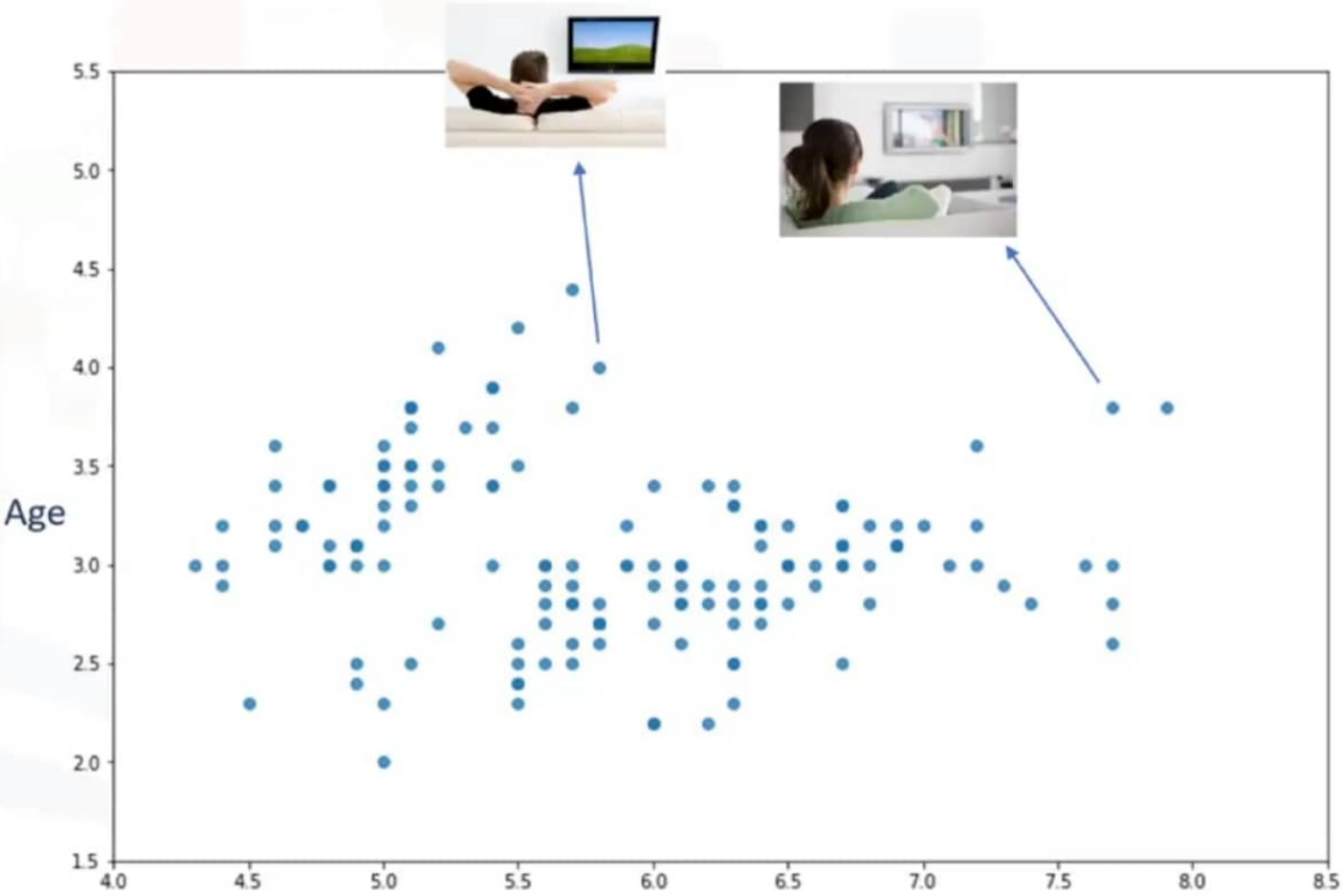
Now, the question is, how can we calculate the dissimilarity or distance of two cases such as two customers? Assume that we have two customers, we will call them Customer one and two. Let's also assume that we have only one feature for each of these two customers and that feature is age.

We can easily use a specific type of Minkowski distance to calculate the distance of these two customers. Indeed, it is the Euclidean distance. Distance of x1 from x2 is root of 34 minus 30_2 which is four. What about if we have more than one feature, for example age and income.

For example, if we have income and age for each customer, we can still use the same formula but this time

How does k-Means clustering work?

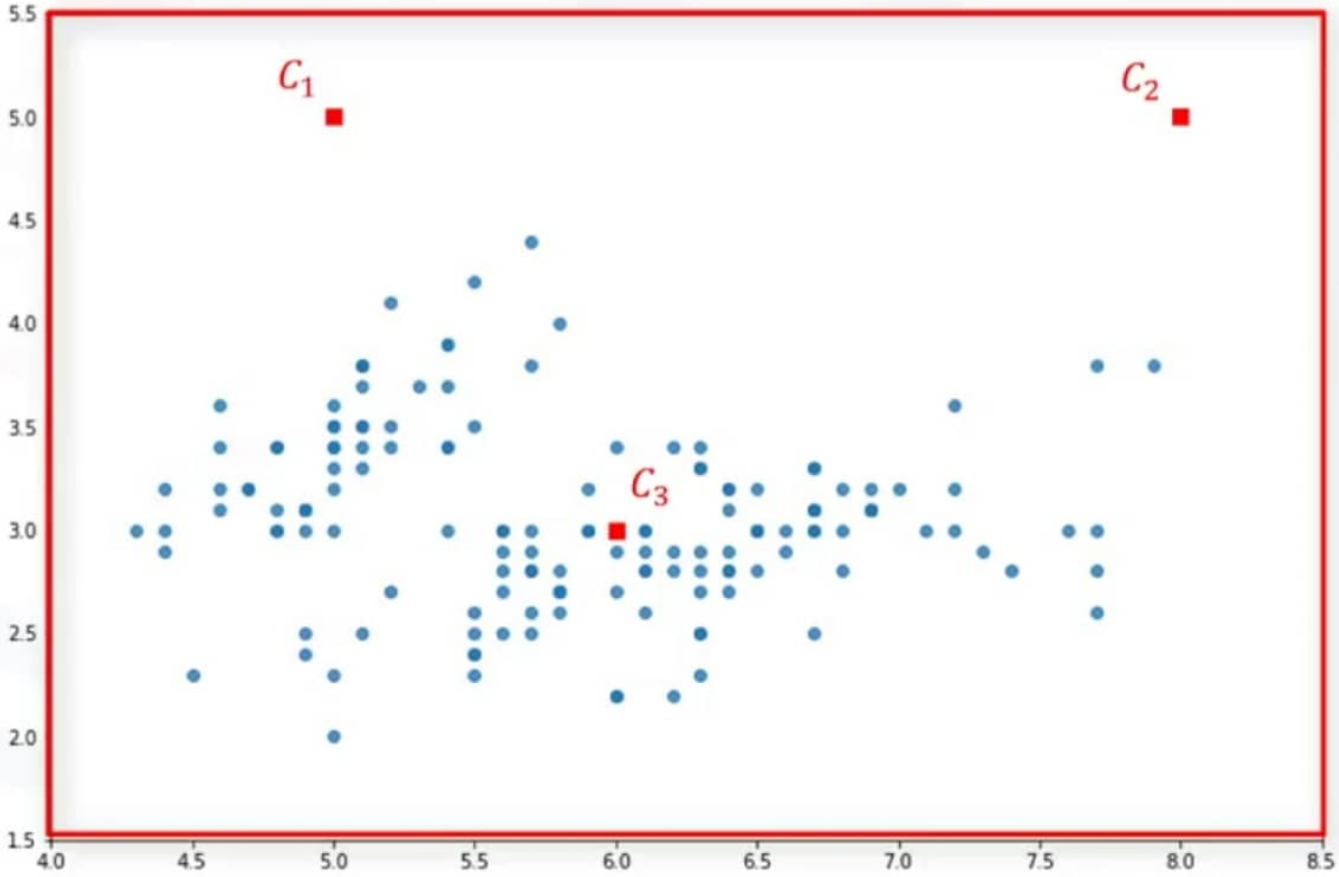
Customer ID	Age	Income
1	3	4
2	2	6
3	3.5	2
...	...	..



k-Means clustering – initialize k

1) Initialize k=3 centroids randomly

C_1 = [8., 5.]
C_2 = [5., 5.]
C_3 = [6., 3.]

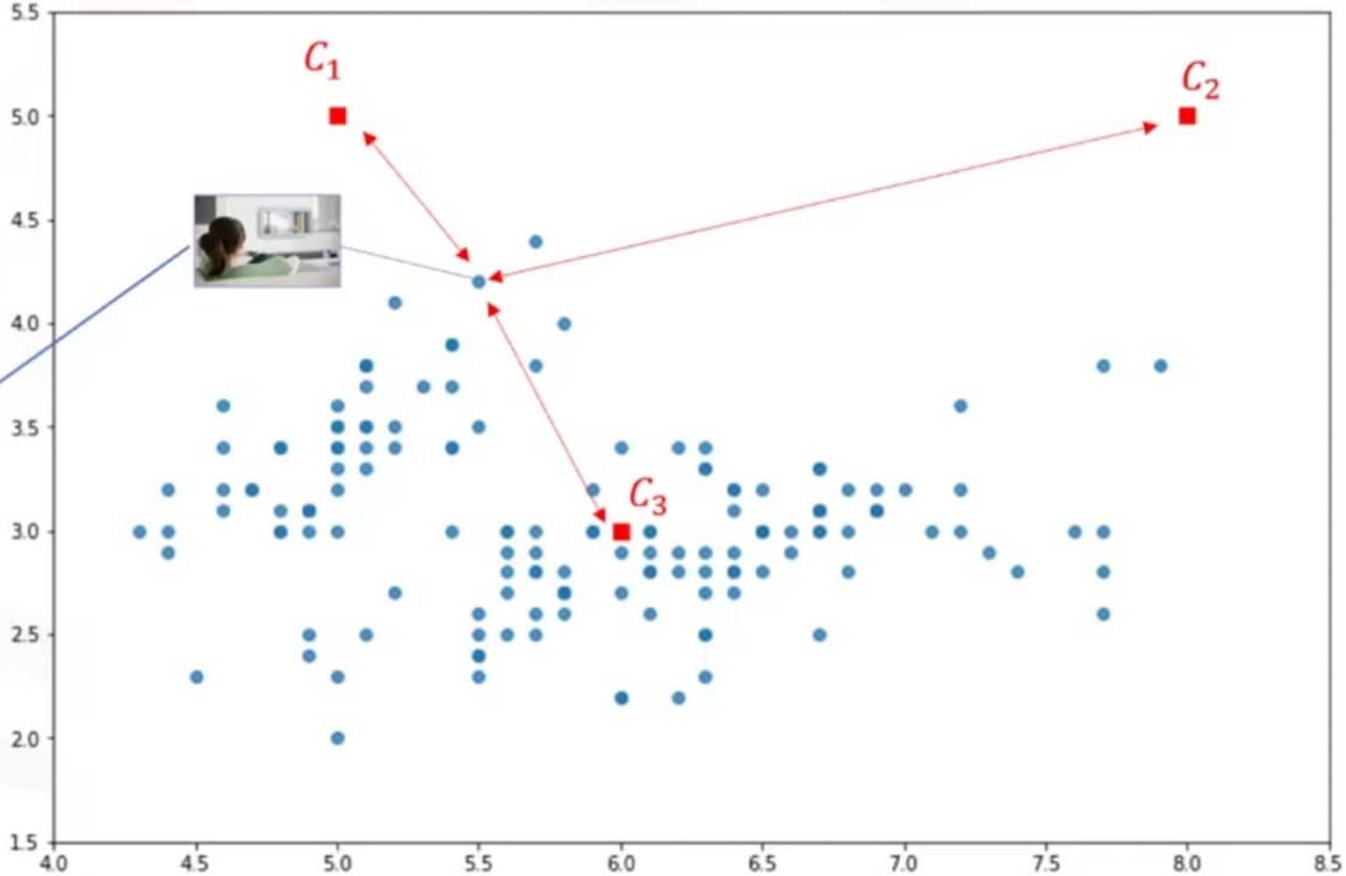


K-Means clustering – calculate the distance

2) Distance calculation

$$
\begin{array}{ccc}
C_1 & C_2 & C_3 \\
\end{array}
$$

$$
\begin{bmatrix}
d(p1,c1) & d(p1,c2) & d(p1,c3) \\
d(p2,c1) & d(p2,c2) & d(p2,c3) \\
d(p3,c1) & d(p3,c2) & d(p3,c3) \\
d(p4,c1) & d(p4,c2) & d(p4,c3) \\
d(p\ ...,c1) & d(p..,c2) & d(p..,c3) \\
d(pn,c1) & d(pn,c2) & d(pn,c3) \\
d(p..,c1) & d(p..,c2) & d(p..,c3) \\
d(p\ ...,c1) & d(p..,c2) & d(p..,c3) \\
d(pn,c1) & d(pn,c2) & d(pn,c3) \\
d(p..,c1) & d(p..,c2) & d(p..,c3) \\
d(p\ ...,c1) & d(p..,c2) & d(p..,c3) \\
d(pn,c1) & d(pn,c2) & d(pn,c3) \\
d(p..,c1) & d(p..,c2) & d(p..,c3) \\
d(p\ ...,c1) & d(p..,c2) & d(p..,c3) \\
d(pn,c1) & d(pn,c2) & d(pn,c3)
\end{bmatrix}
$$

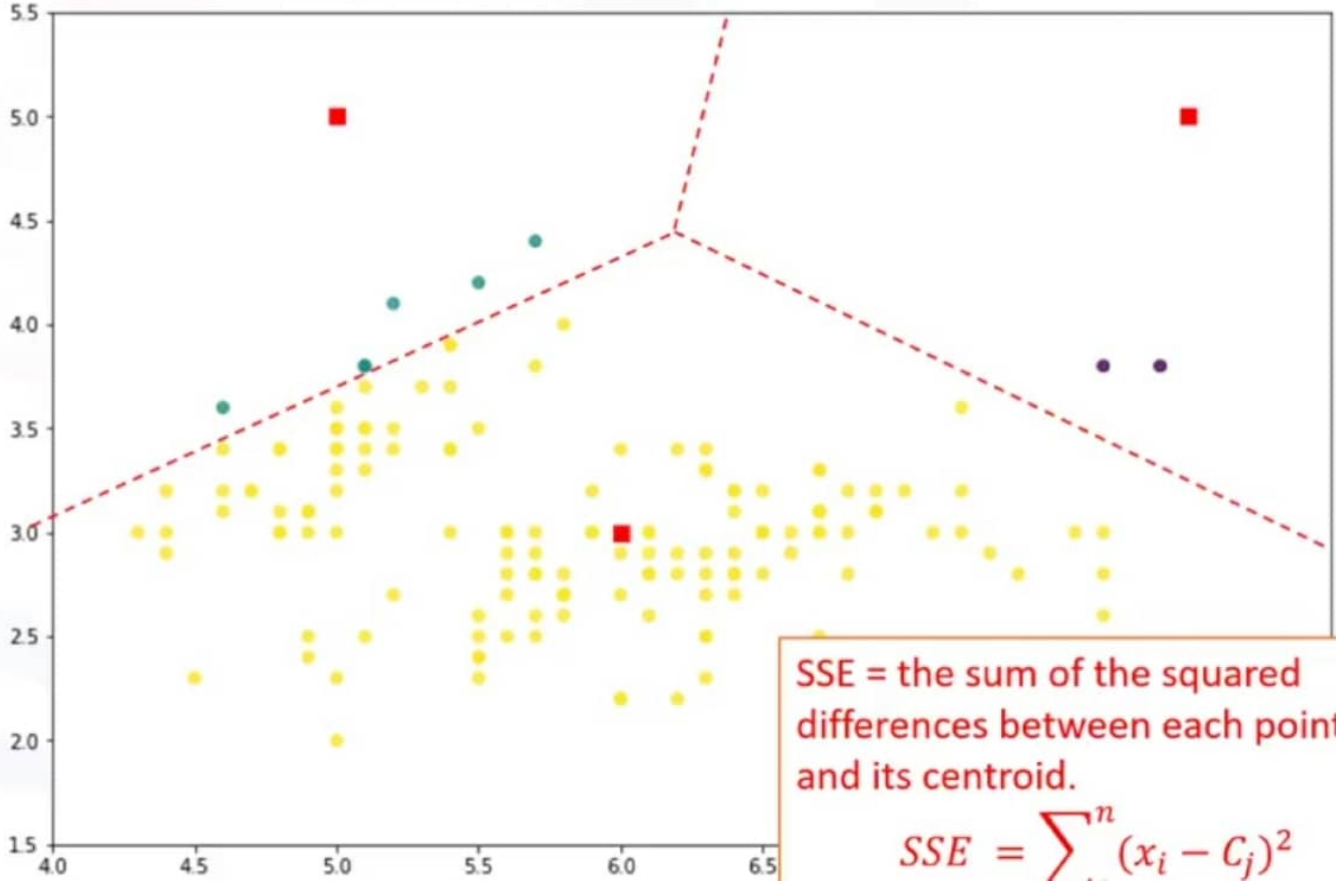


k-Means clustering – assign to centroid

3) Assign each point to the closest centroid

$$
\begin{array}{ccc} C_1 & C_2 & C_3 \end{array}
$$

$$
\begin{bmatrix}
d(p1,c1) & d(p1,c2) & d(p1,c3) \\
d(p2,c1) & d(p2,c2) & d(p2,c3) \\
d(p3,c1) & d(p3,c2) & d(p3,c3) \\
d(p4,c1) & d(p4,c2) & d(p4,c3) \\
d(p\,...,c1) & d(p..,c2) & d(p..,c3) \\
d(pn,c1) & d(pn,c2) & d(pn,c3) \\
d(p..,c1) & d(p..,c2) & d(p..,c3) \\
d(p\,...,c1) & d(p..,c2) & d(p..,c3) \\
d(pn,c1) & d(pn,c2) & d(pn,c3) \\
d(p..,c1) & d(p..,c2) & d(p..,c3) \\
d(p\,...,c1) & d(p..,c2) & d(p..,c3) \\
d(pn,c1) & d(pn,c2) & d(pn,c3) \\
d(p..,c1) & d(p..,c2) & d(p..,c3) \\
d(p\,...,c1) & d(p..,c2) & d(p..,c3) \\
d(pn,c1) & d(pn,c2) & d(pn,c3)
\end{bmatrix}
$$



SSE = the sum of the squared differences between each point and its centroid.

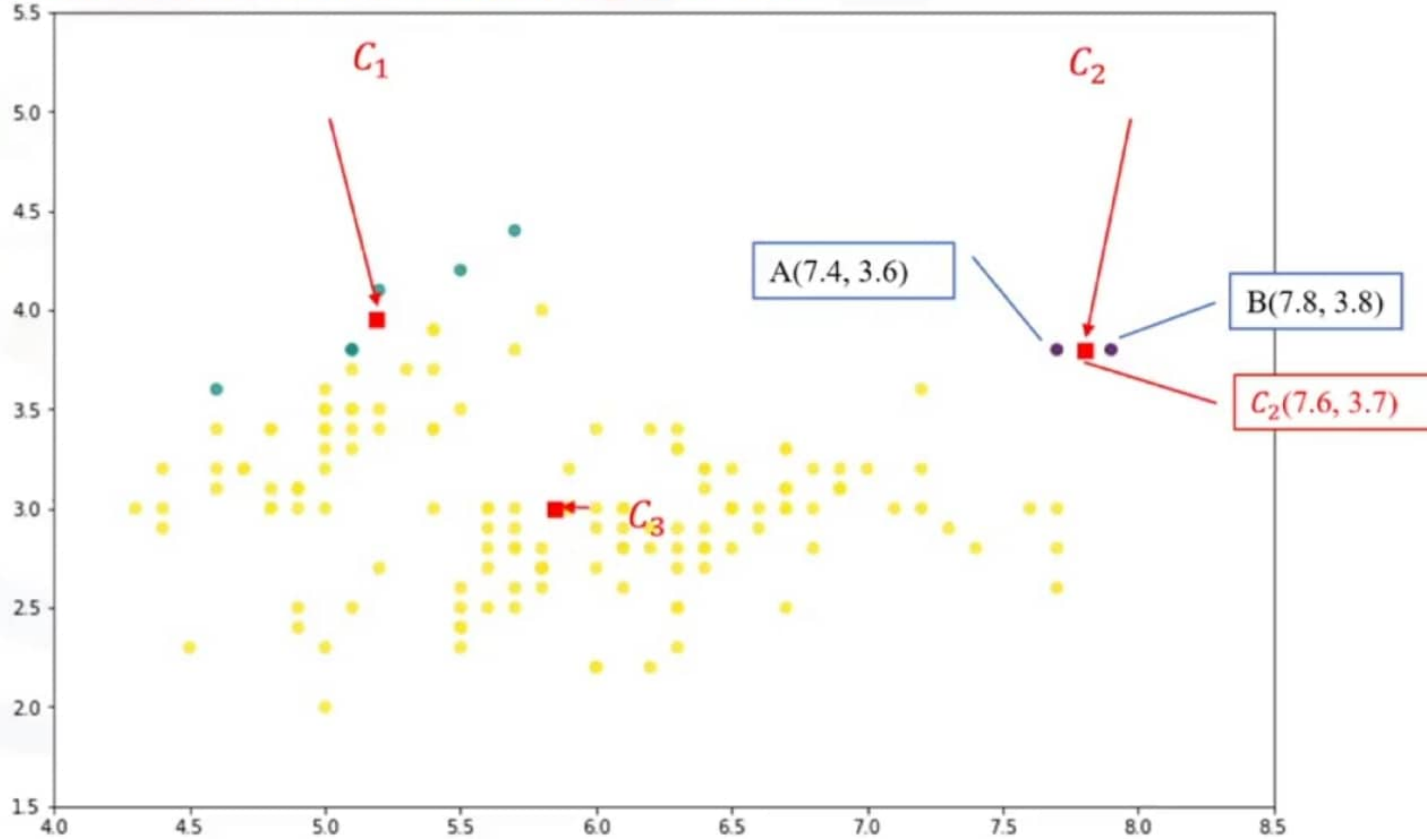
$$SSE = \sum^{n} (x_i - C_j)^2$$

Scanned with CamScanner

k-Means clustering – compute new centroids

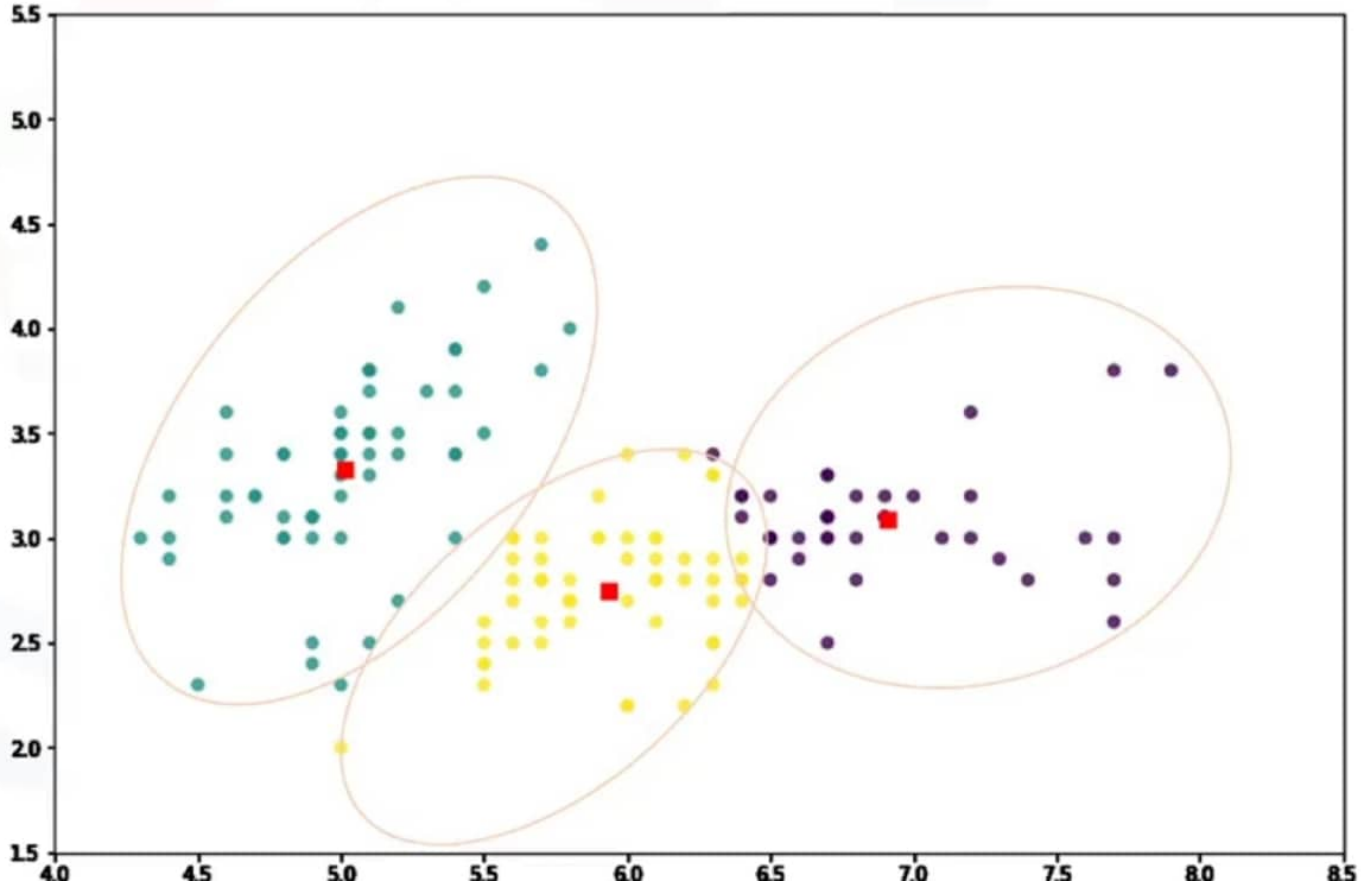
4) Compute the new centroids for each cluster.

C_1	C_2	C_3
$d(p1, c1)$	$d(p1, c2)$	$d(p1, c3)$
$d(p2, c1)$	$d(p2, c2)$	$d(p2, c3)$
$d(p3, c1)$	$d(p3, c2)$	$d(p3, c3)$
$d(p4, c1)$	$d(p4, c2)$	$d(p4, c3)$
$d(p\ ..., c1)$	$d(p.., c2)$	$d(p.., c3)$
$d(pn, c1)$	$d(pn, c2)$	$d(pn, c3)$
$d(p.., c1)$	$d(p.., c2)$	$d(p.., c3)$
$d(p\ ..., c1)$	$d(p.., c2)$	$d(p.., c3)$
$d(pn, c1)$	$d(pn, c2)$	$d(pn, c3)$
$d(p.., c1)$	$d(p.., c2)$	$d(p.., c3)$
$d(p\ ..., c1)$	$d(p.., c2)$	$d(p.., c3)$
$d(pn, c1)$	$d(pn, c2)$	$d(pn, c3)$
$d(p.., c1)$	$d(p.., c2)$	$d(p.., c3)$
$d(p\ ..., c1)$	$d(p.., c2)$	$d(p.., c3)$
$d(pn, c1)$	$d(pn, c2)$	$d(pn, c3)$



k-Means clustering – repeat

5) Repeat until there are no more changes.

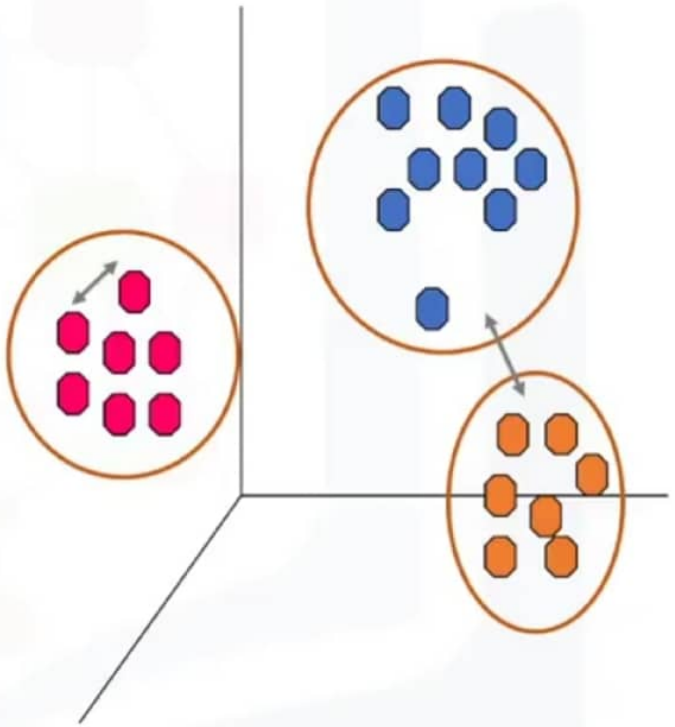


k-Means clustering algorithm

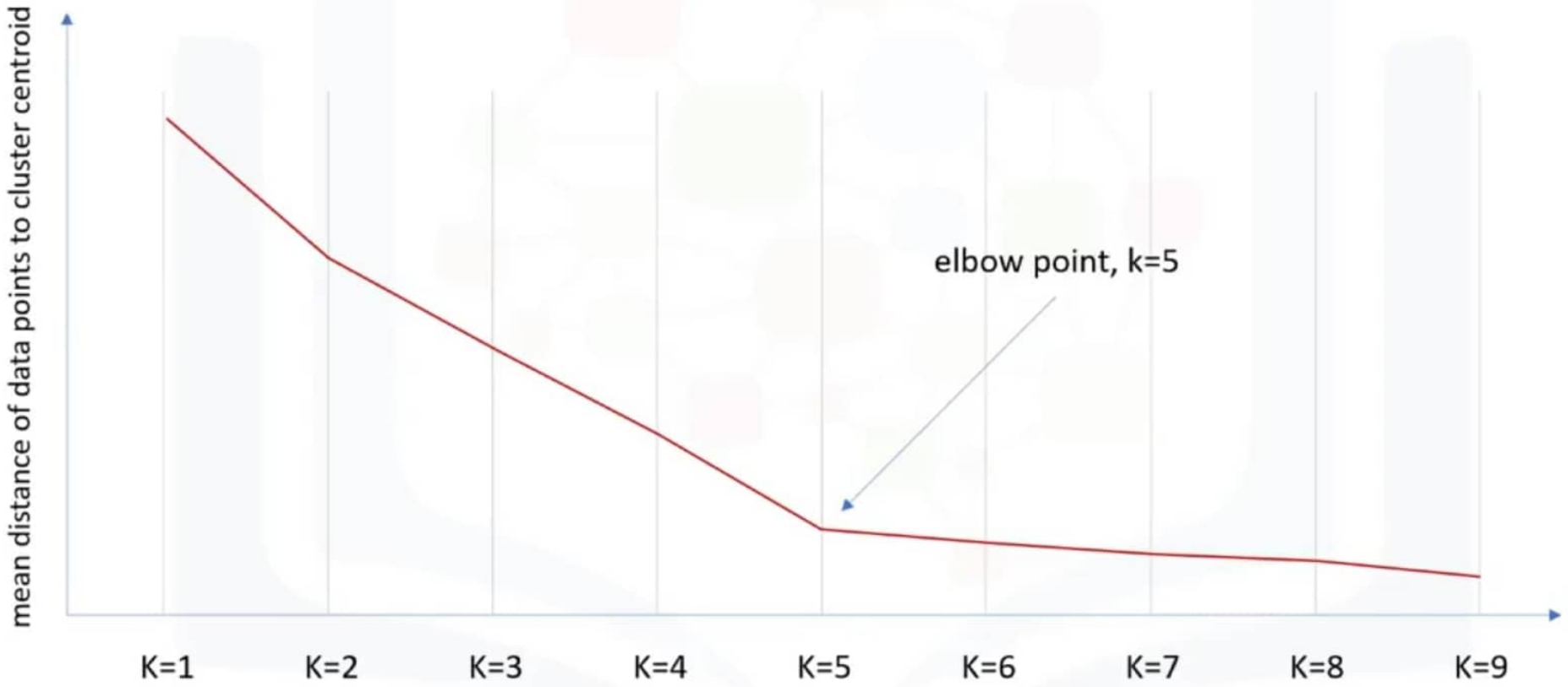
1. Randomly placing *k* centroids, one for each cluster.
2. Calculate the distance of each point from each centroid.
3. Assign each data point (object) to its closest centroid, creating a cluster.
4. Recalculate the position of the *k* centroids.
5. Repeat the steps 2-4, until the centroids no longer move.

k-Means accuracy

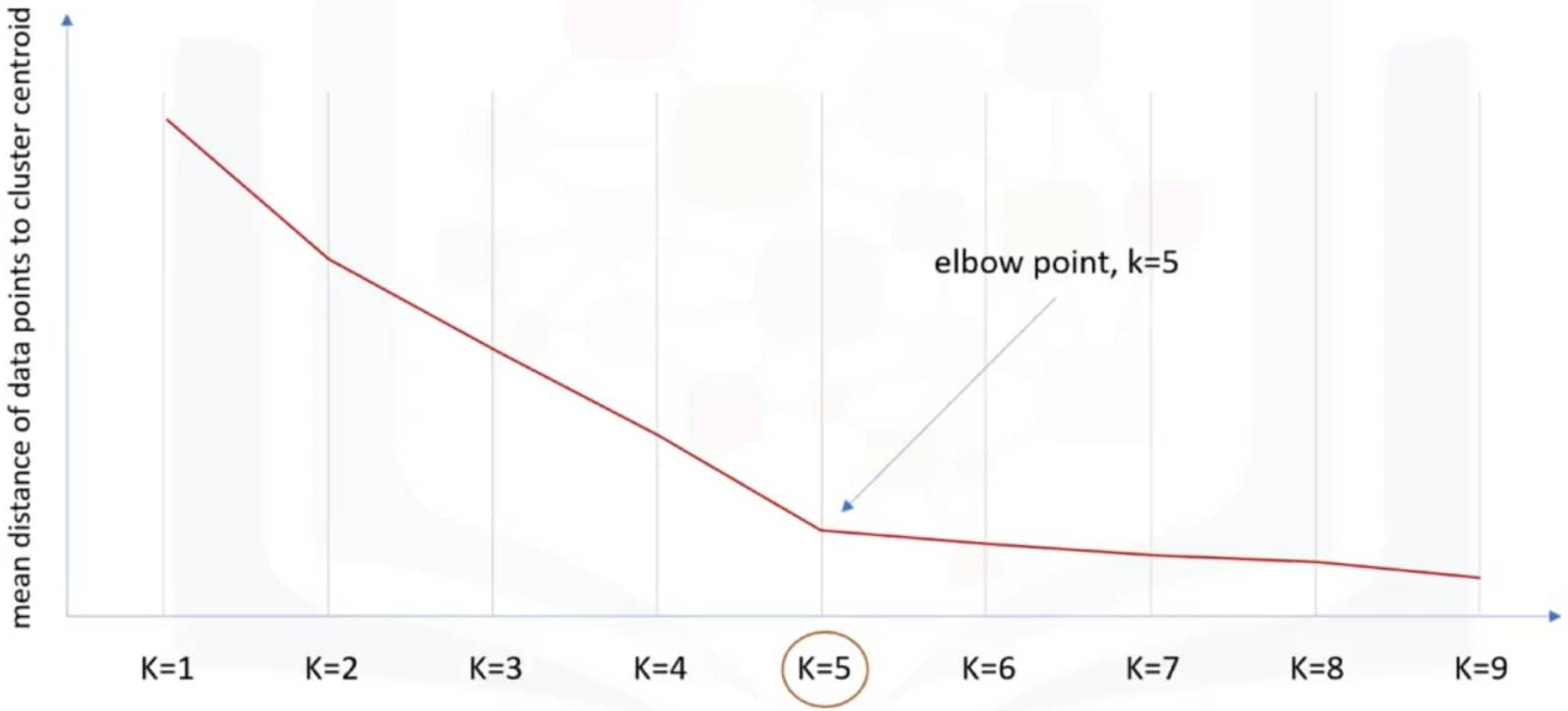
- External approach
  - Compare the clusters with the ground truth, if it is available.
- Internal approach
  - Average the distance between data points within a cluster.



Choosing k



Choosing k

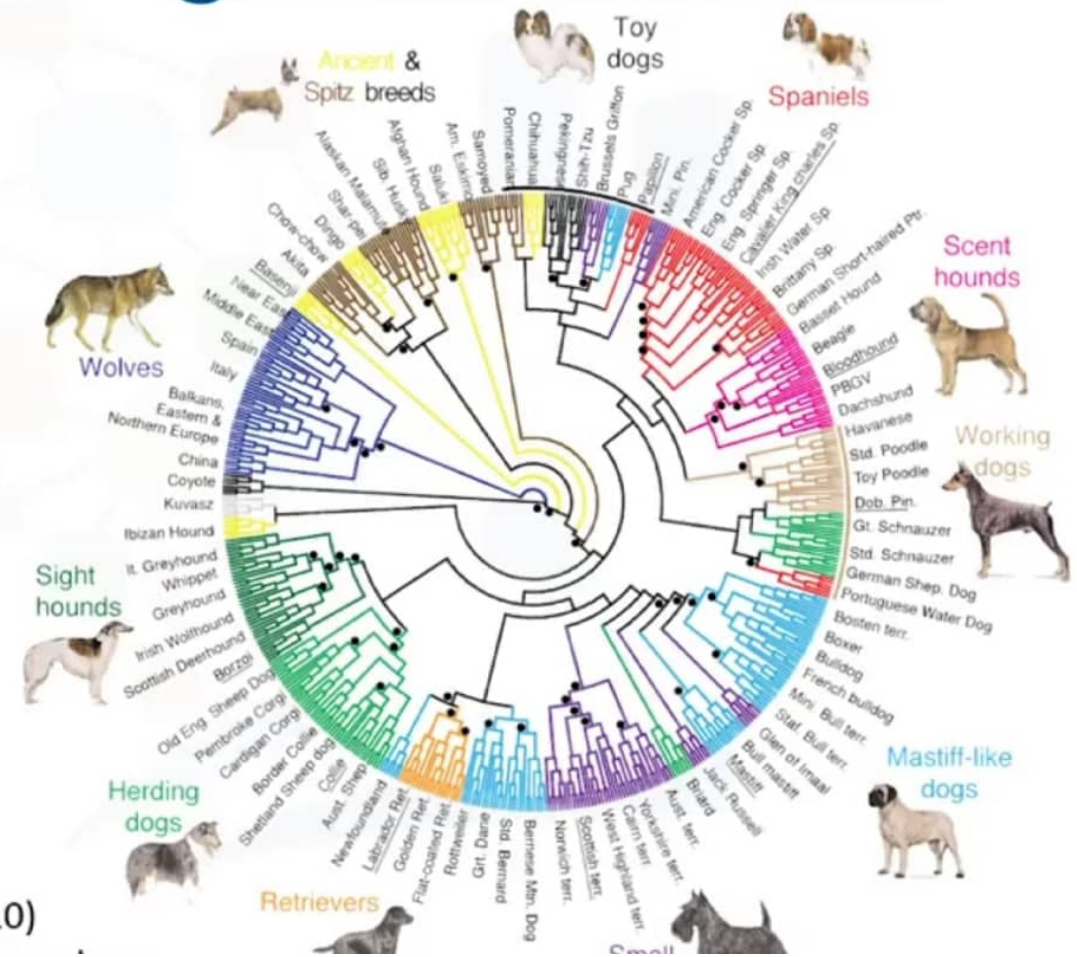


k-Means recap

- Med and Large sized databases (*Relatively efficient*)
- Produces sphere-like clusters
- Needs number of clusters (k)

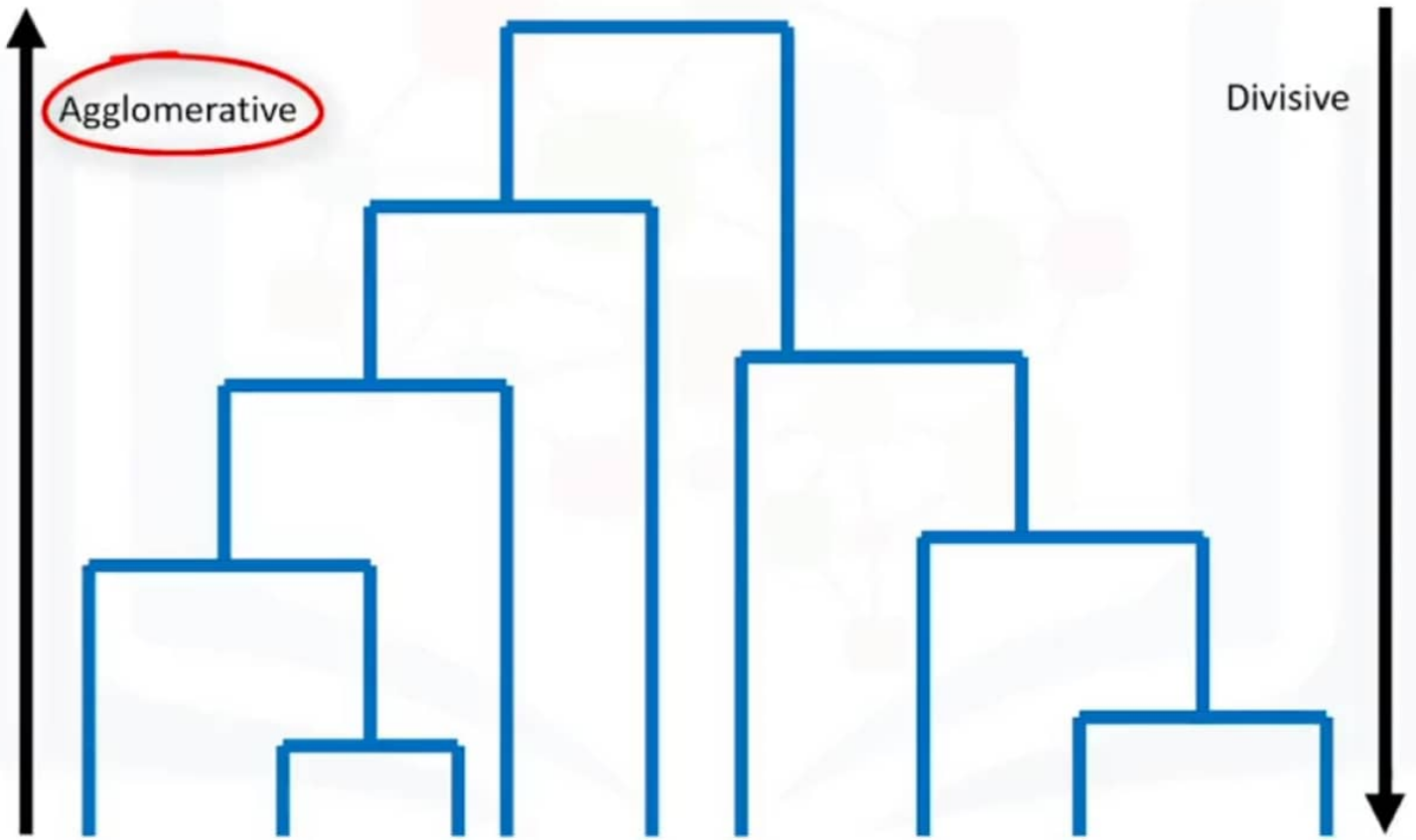
Hierarchical clustering

Hierarchical clustering algorithms build a hierarchy of clusters where each node is a cluster consists of the clusters of its daughter nodes.

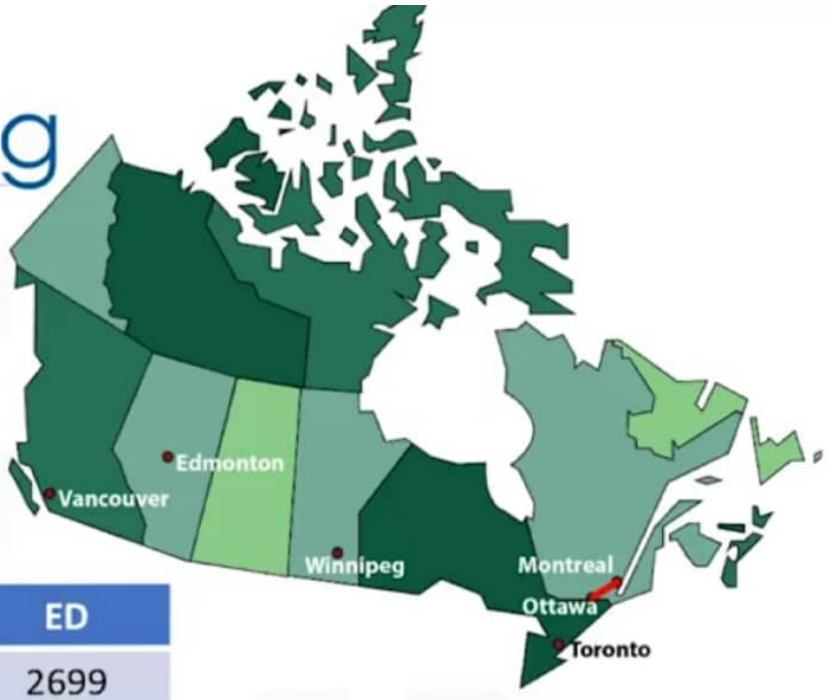


Source: von Holdt B.M. et al. (2010)

Hierarchical clustering



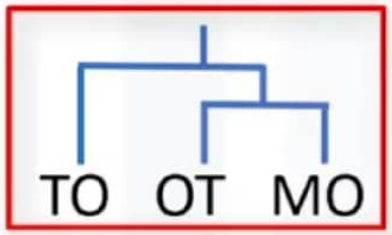
Agglomerative clustering



TO OT MO VA ED WI

	TO	OT	VA	MO	WI	ED
TO		351	3363	505	1510	2699
OT			3543	167	1676	2840
VA				3690	1867	819
MO					1824	2976
WI						1195
ED						

Agglomerative clustering

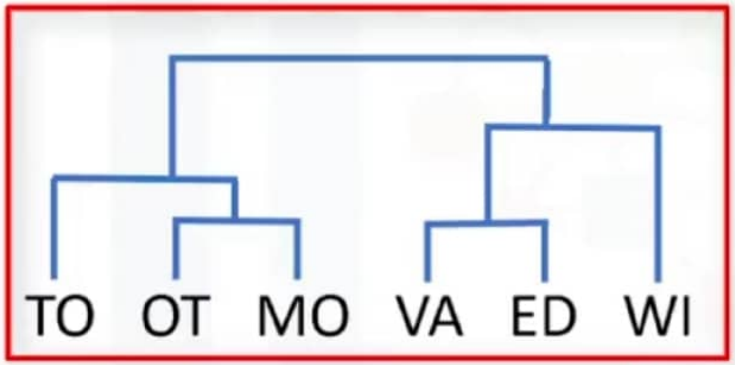


	TO	OT/MO	VA	WI	ED
TO		351	3363	1510	2699
OT/MO			3543	1676	2840
VA				1867	819
WI					1195
ED					

Agglomerative clustering

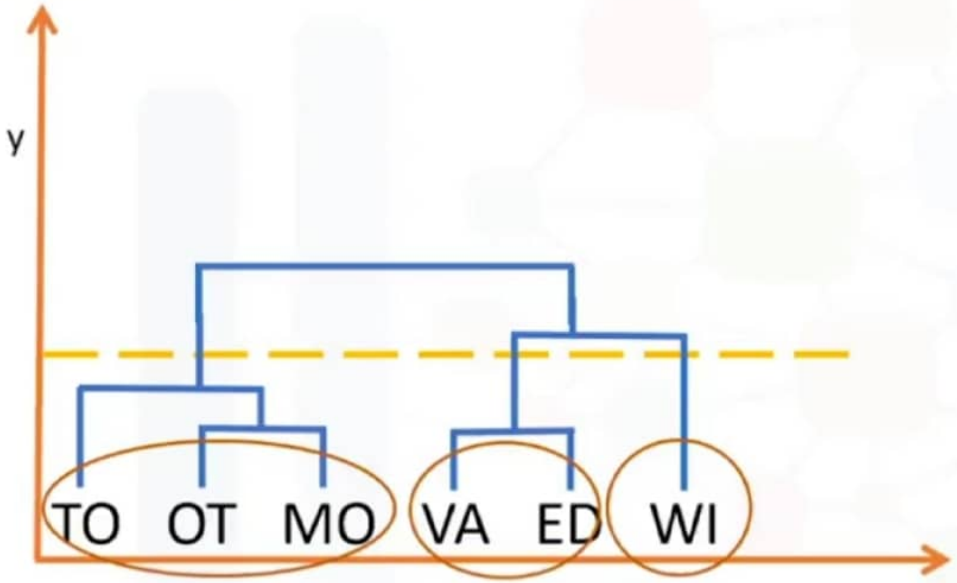
	TO/OT/MO	VA/ED	WI
TO/OT/MO		2840	1676
VA/ED			1667
WI			

Hierarchical clustering



	TO/OT/MO	VA/ED/WI
TO/OT/MO		1676
VA/ED/WI		

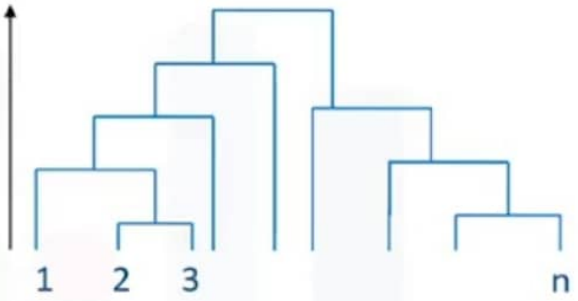
Hierarchical clustering



Dendrogram

Agglomerative algorithm

1. Create *n* clusters, one for each data point
2. Compute the Proximity Matrix
3. **Repeat**
    i. Merge the two closest clusters
    ii. Update the proximity matrix
4. **Until** only a single cluster remains



$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \dots & \dots & 0 \end{bmatrix}$$

Similarity/Distance

Patient 1		
Age	BMI	BP
54	190	120

Patient 2		
Age	BMI	BP
50	200	125

Dis (p1,p2)

$$= \sqrt{\sum_{i=0}^{n}(x_i - y_i)^2}$$

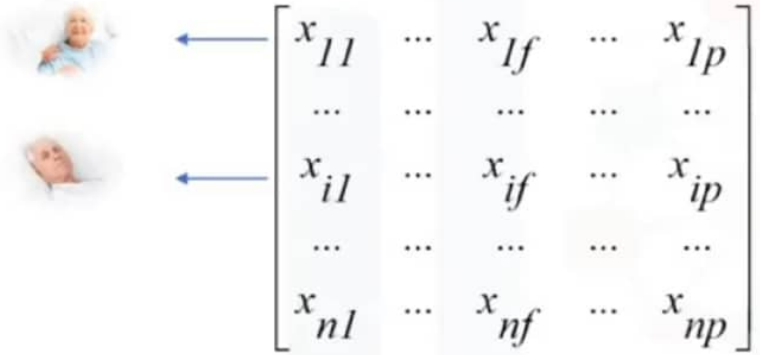
$$= \sqrt{(54-50)^2+(190-200)^2+(120-125)^2}$$

$$= 11.87$$

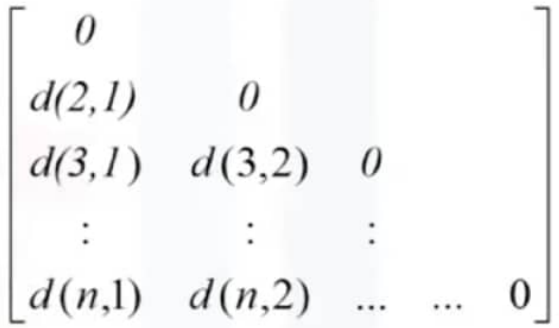
We can use different distance measurements to calculate the proximity matrix.

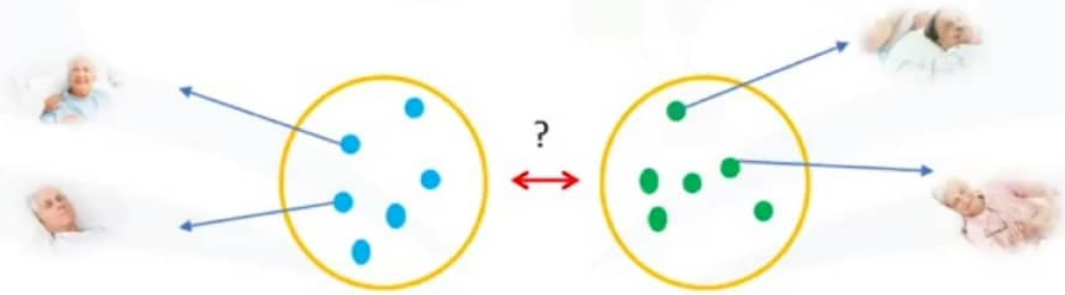
Similarity/Distance

Dataset



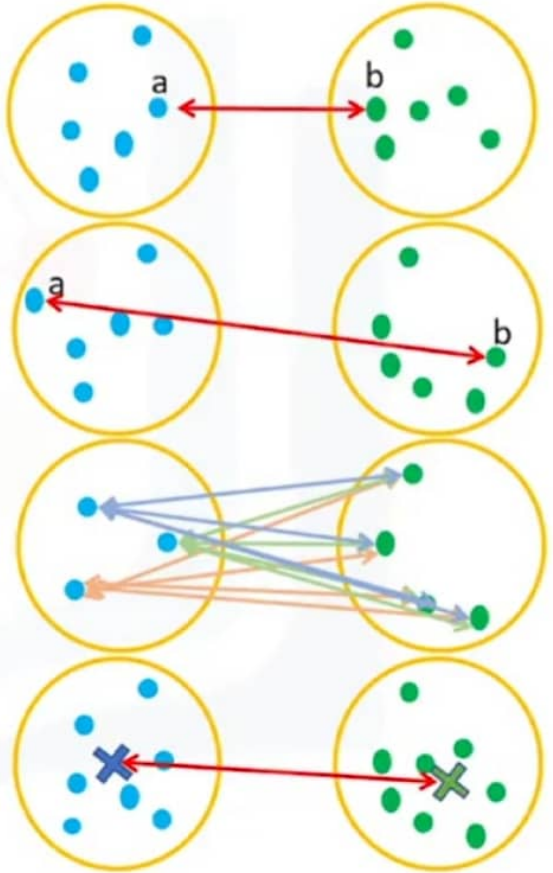
Dissimilarity matrix





Distance between clusters

- Single-Linkage Clustering
  - Minimum distance between clusters
- Complete-Linkage Clustering
  - Maximum distance between clusters
- Average Linkage Clustering
  - Average distance between clusters
- Centroid Linkage Clustering
  - Distance between cluster centroids



Advantages vs. disadvantages

Advantages	Disadvantages
Doesn't required number of clusters to be specified.	Can never undo any previous steps throughout the algorithm.
Easy to implement.	Generally has long runtimes.
Produces a dendrogram, which helps with understanding the data.	Sometimes difficult to identify the number of clusters by the dendrogram.

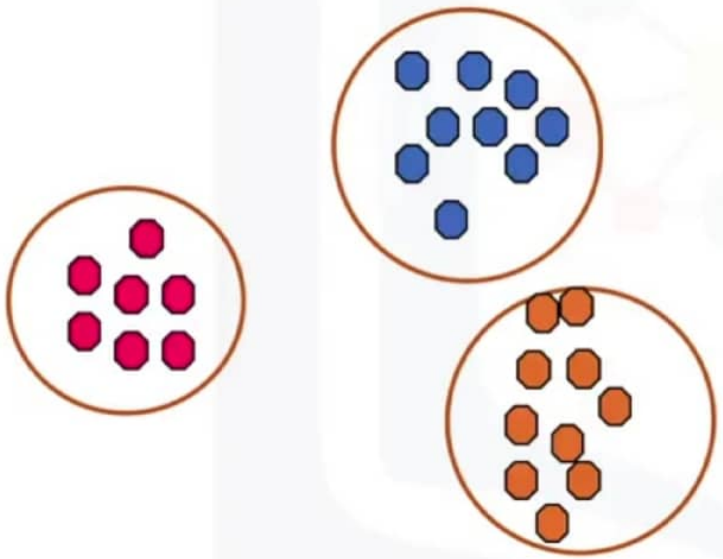
Hierarchical clustering Vs. *K*-means

K-means	Hierarchical Clustering
1. Much more efficient	1. Can be slow for large datasets
2. Requires the number of clusters to be specified	2. Does not require the number of clusters to run
3. Gives only one partitioning of the data based on the predefined number of clusters	3. Gives more than one partitioning depending on the resolution
4. Potentially returns different clusters each time it is run due to random initialization of centroids	4. Always generates the same clusters

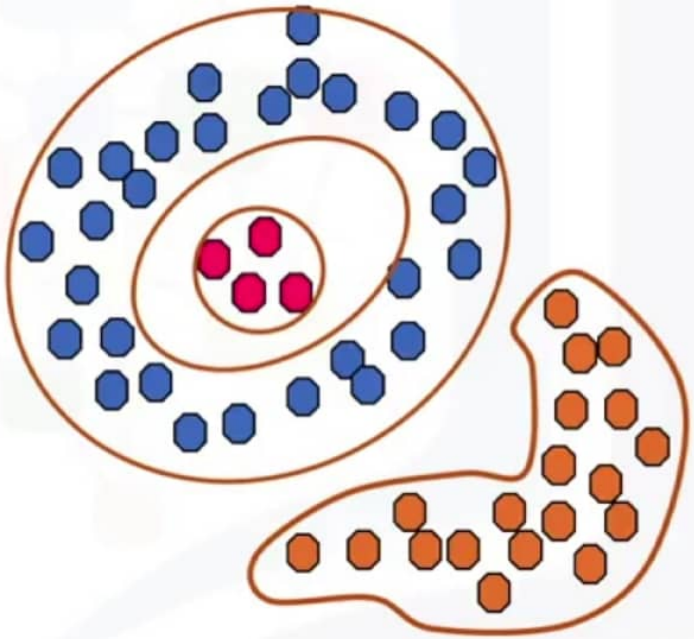
DBSCAN Clustering

Density-based clustering

- Spherical-shape clusters

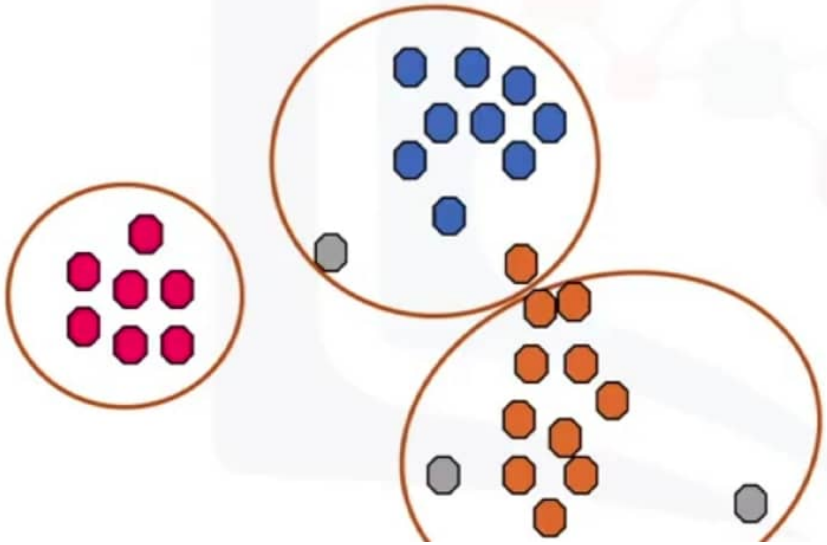


- Arbitrary-shape clusters

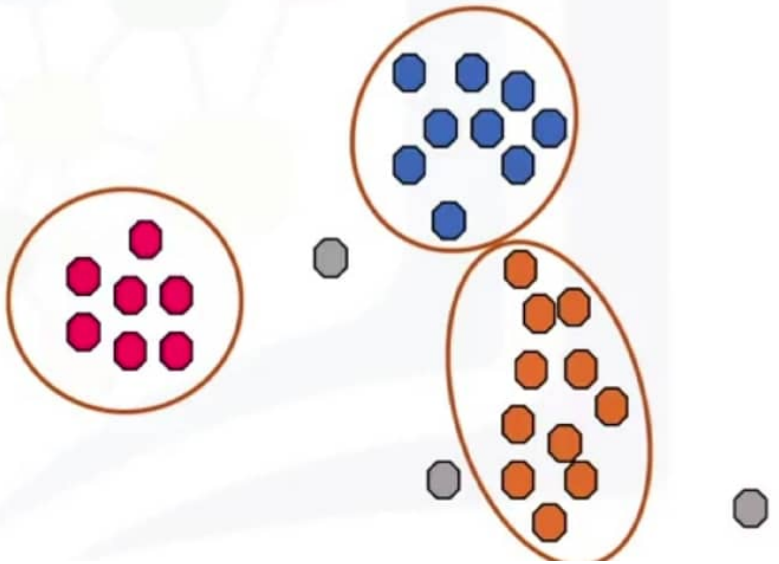


k-Means Vs. density-based clustering

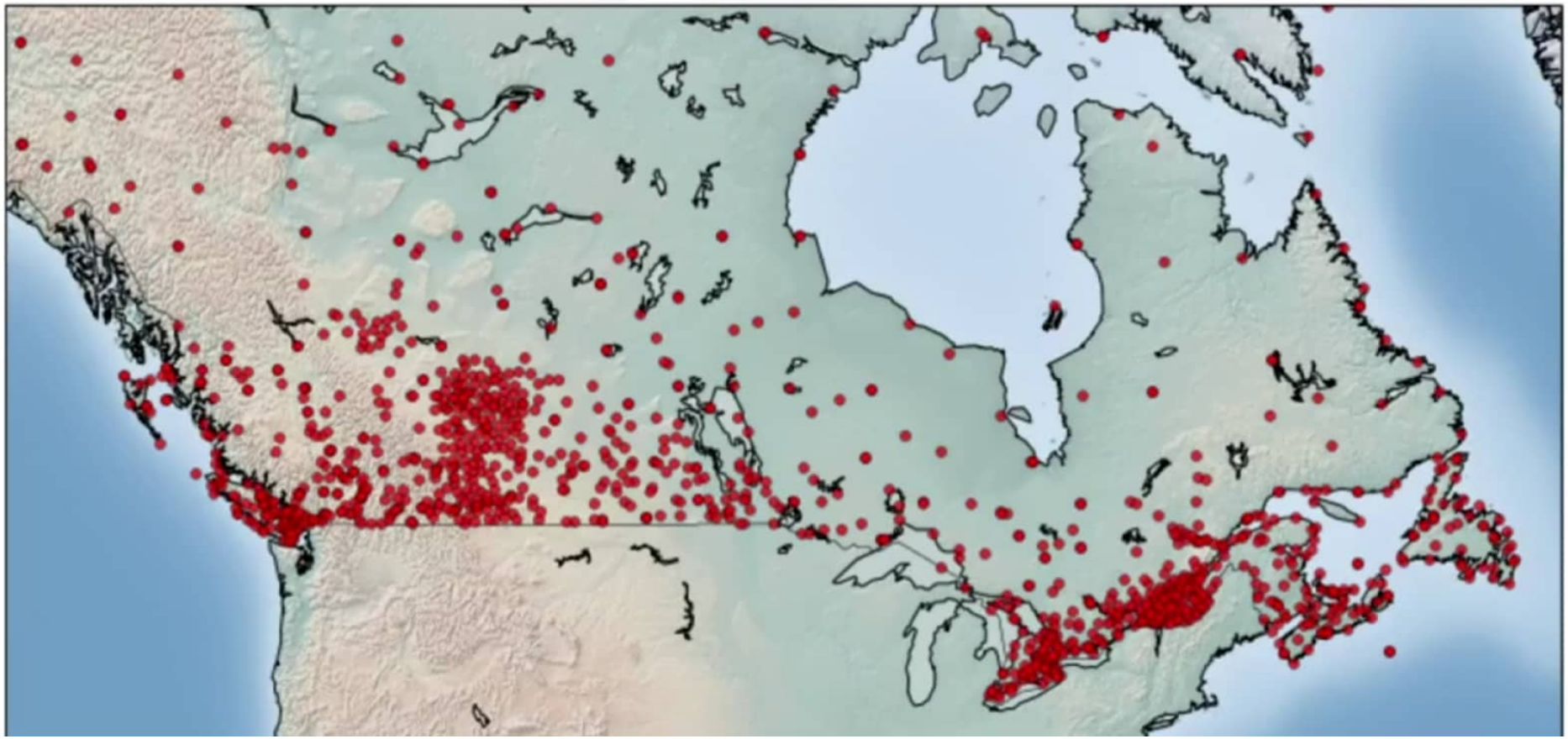
- k-Means assigns all points to a cluster even if they do not belong in any



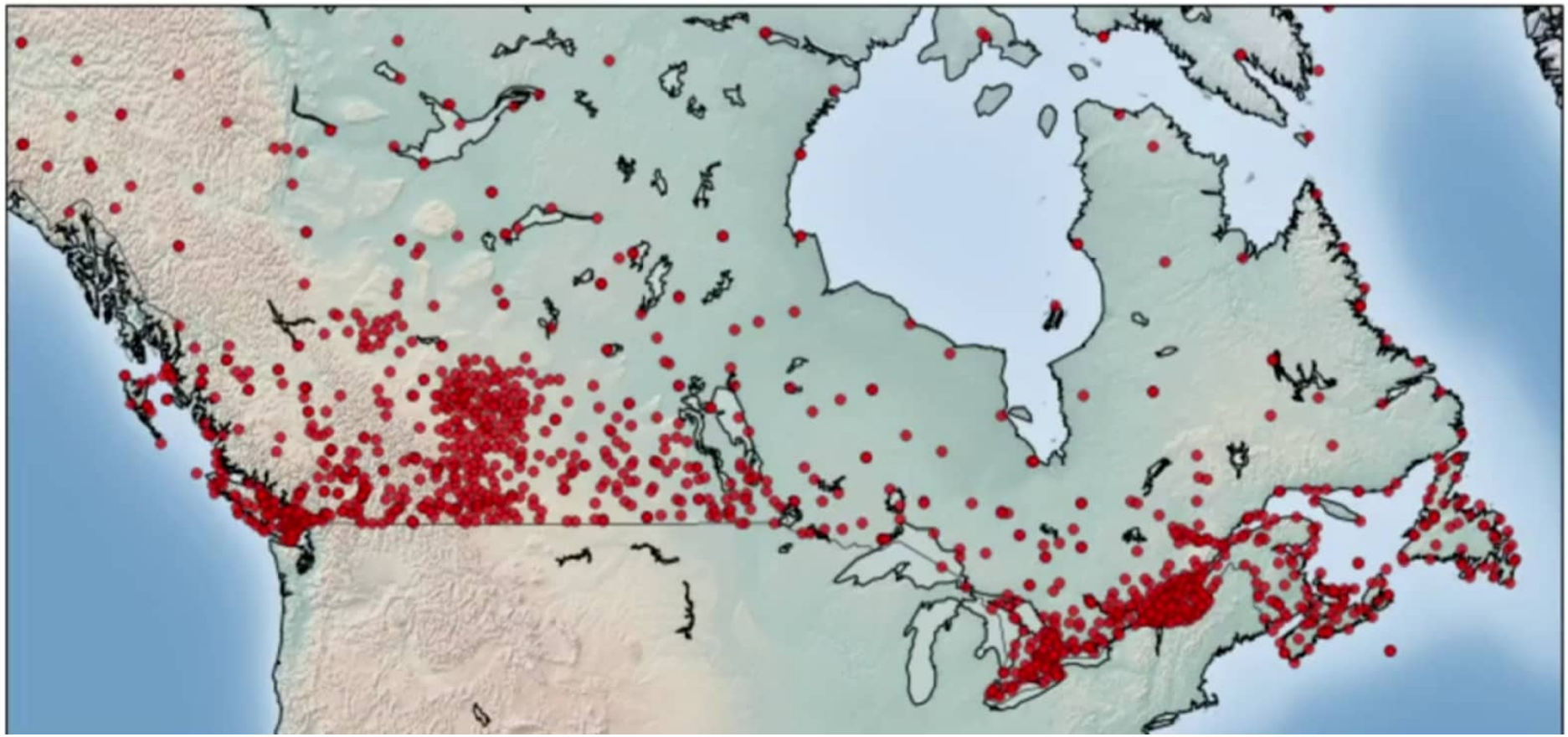
- Density-based Clustering locates regions of high density, and separates outliers



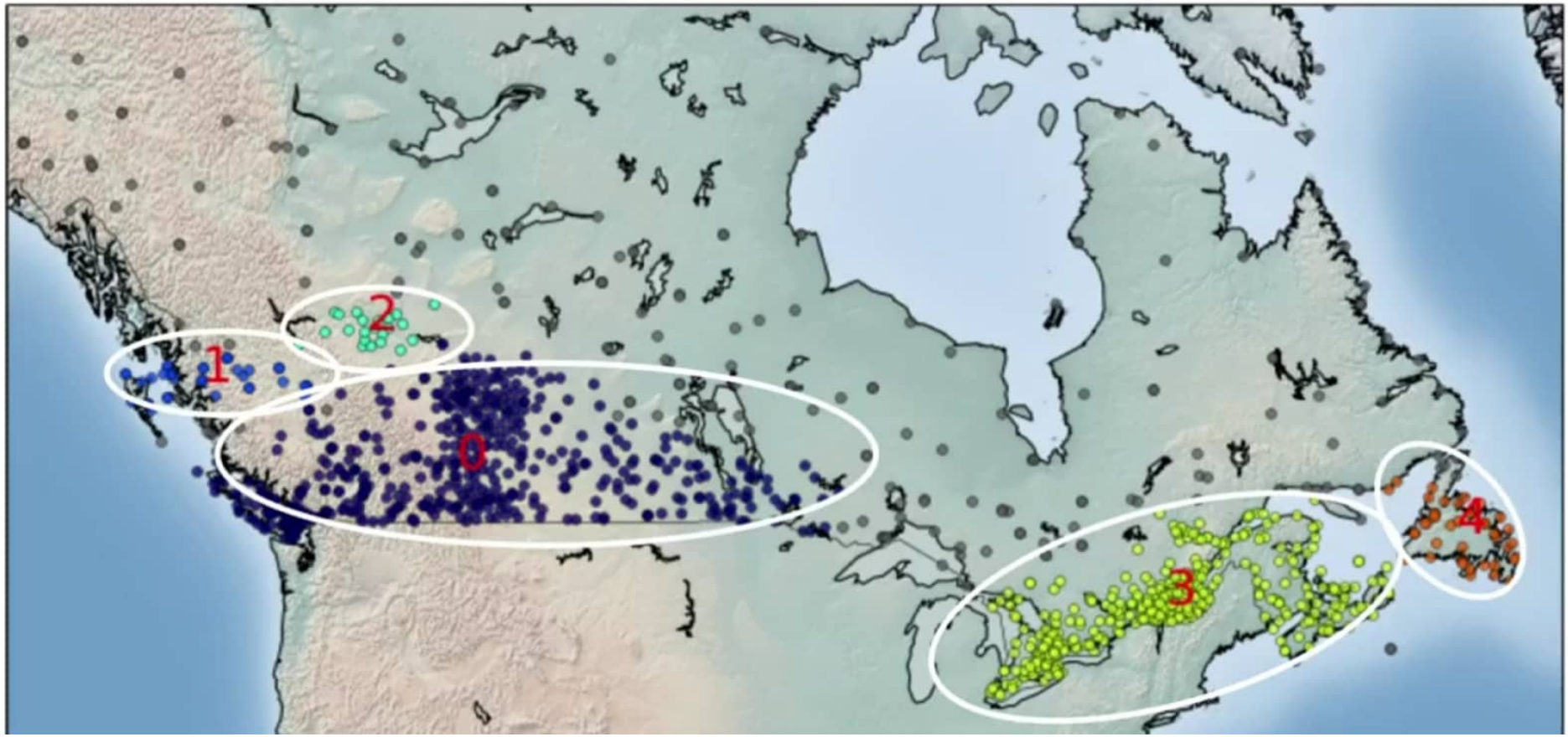
DBSCAN for class identification



DBSCAN for class identification



DBSCAN for class identification

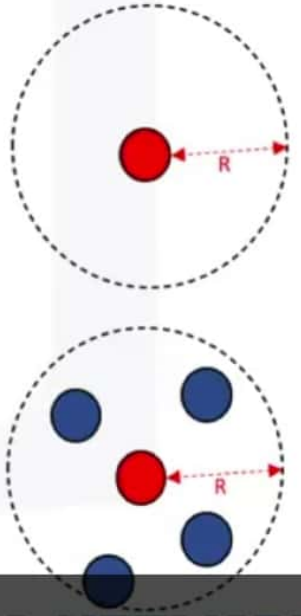


What is DBSCAN?

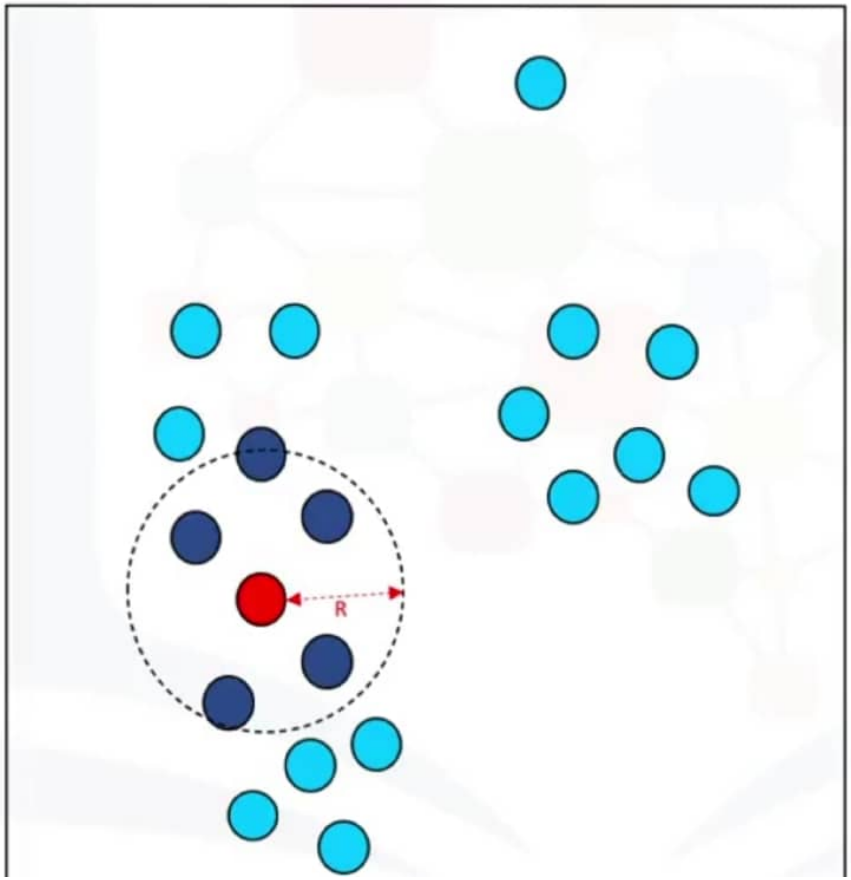
- DBSCAN (**D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise)
  - Is one of the most common clustering algorithms
  - Works based on density of objects

- R (**R**adius of neighborhood)
  - Radius (R) that if includes enough number of points within, we call it a dense area

- M (**M**in number of neighbors)
  - The minimum number of data points we want in a neighborhood to define a cluster

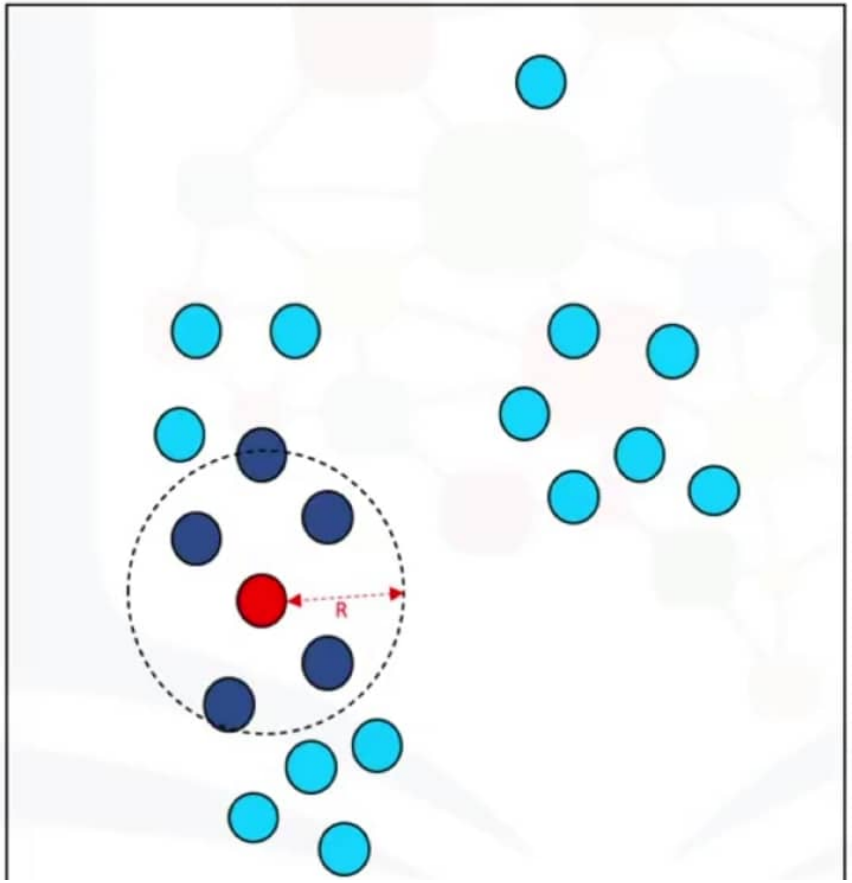


DBSCAN algorithm – core point



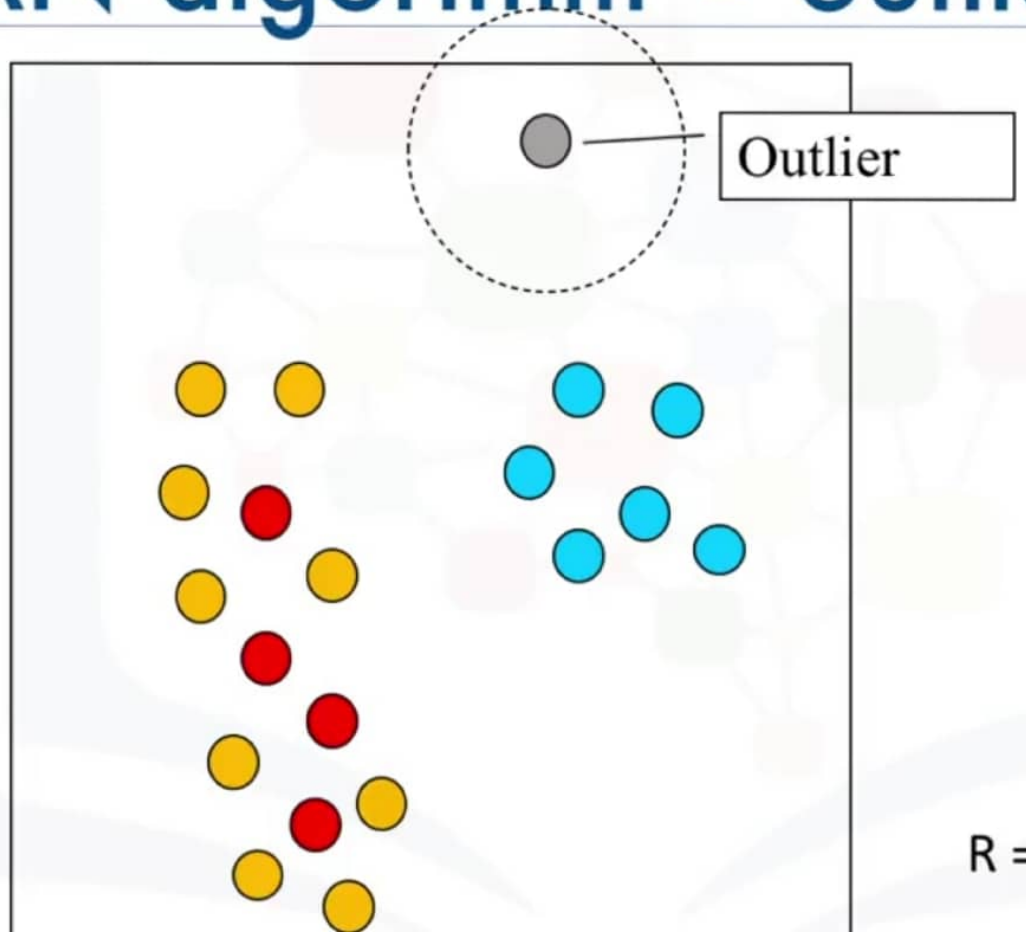
R = 2unit , M = 6

DBSCAN algorithm – core point



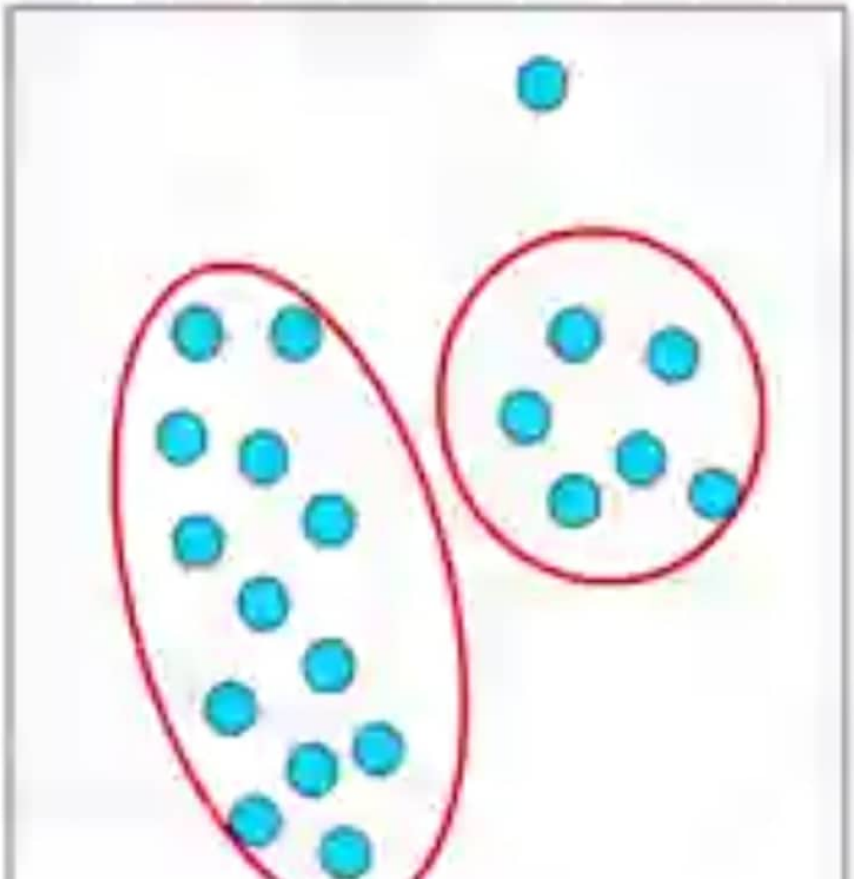
R = 2unit , M = 6

DBSCAN algorithm – outliers



R = 2unit , M = 6

Advantages of DBSCAN



1. Arbitrarily shaped clusters
2. Robust to outliers
3. Does not require specification of the number of clusters